# Analyse von Ansätzen zur Beschleunigung von SAT - Lösern durch dedizierte Hardware Komponenten

E. Zenker

24. Februar 2011

# Gliederung

1. Satisfiability Testing
2. FPGAs
3. Aktuelle Hardware SAT Solver

# Satisfiability Testing - SAT

- Variablen und Domäne der Aussagenlogik

  $V = \{ 1, 2, ..., n \}$

  $D = \{\top, \bot\}$

- Operationen

  Negation $\neg$, Konjunktion $\wedge$, Disjunktion $\vee$

- Literale sind Variablen oder negierte Variablen

- Klausel

  Distjunktion von Literalen $C_1 = [\ 1, \neg 2, 3\ ]$

- Formel

  Konjunktion von Klauseln $F = \langle C_1, C_2, ... \rangle$

# Satisfiability Testing - SAT

- Gegeben: Eine Konjunktion von Klauseln (Formel)
- Gesucht: Ein Model für die Formel, wenn vorhanden

$$F = \langle [1, 3], [\neg 2, \neg 5, \neg 6], [\neg 1, \neg 4, 6], [\neg 1, \neg 2, \neg 4, 5], [\neg 1, 2] \rangle$$

- Frage: Wie kann ein solches Model gefunden werden ?

# Lösen des SAT Problems

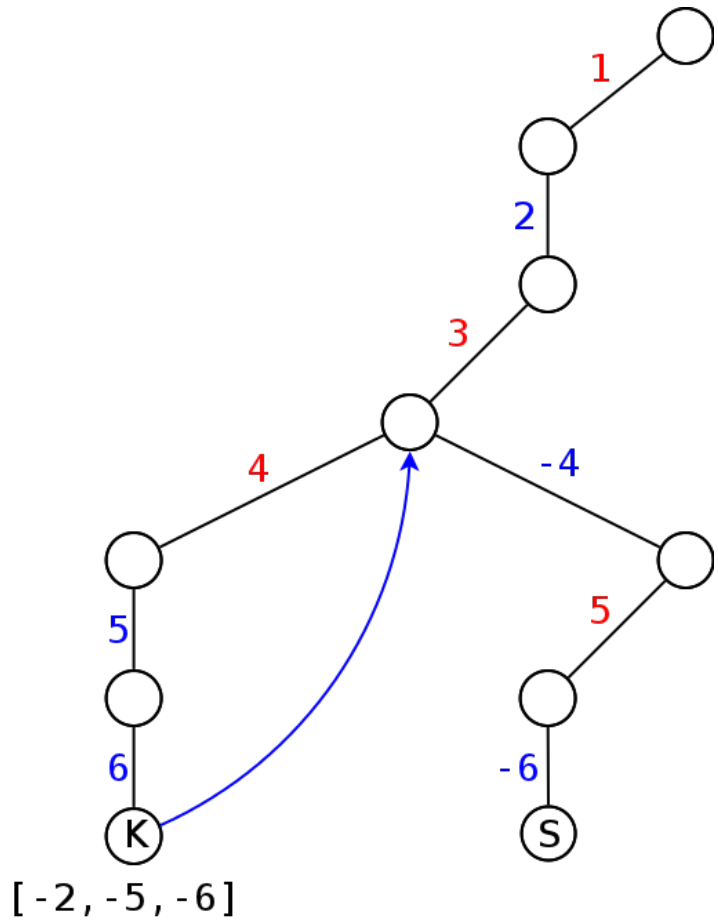
## Davis Putman Logeman Loveland (DPLL)

1. Entscheidung für eine freie Variable
   - Sollte keine freie Variable mehr vorhanden sein wird SAT zurückgegeben
2. Unit Propagation
   - Im Fall eines Konflikts:
     - keine Entscheidungsvariable dann UNSAT zurückgegeben
     - Ändern der letzten Entscheidung (die Selbe nicht zweimal)
     - Pfad bis zu dieser Entscheidung rückgängig machen
     - Algorithmus bei 2. fortführen
   - Bei keinem Konflikt wieder bei 1. beginnen

# Lösen des SAT Problems

## Davis Putman Logeman Loveland (DPLL)

- F=⟨ [1, 3], [¬2, ¬5, ¬6], [¬1, ¬4, 6], [¬1,¬2,¬4, 5], [¬1, 2]⟩



# Lösen des SAT Problems

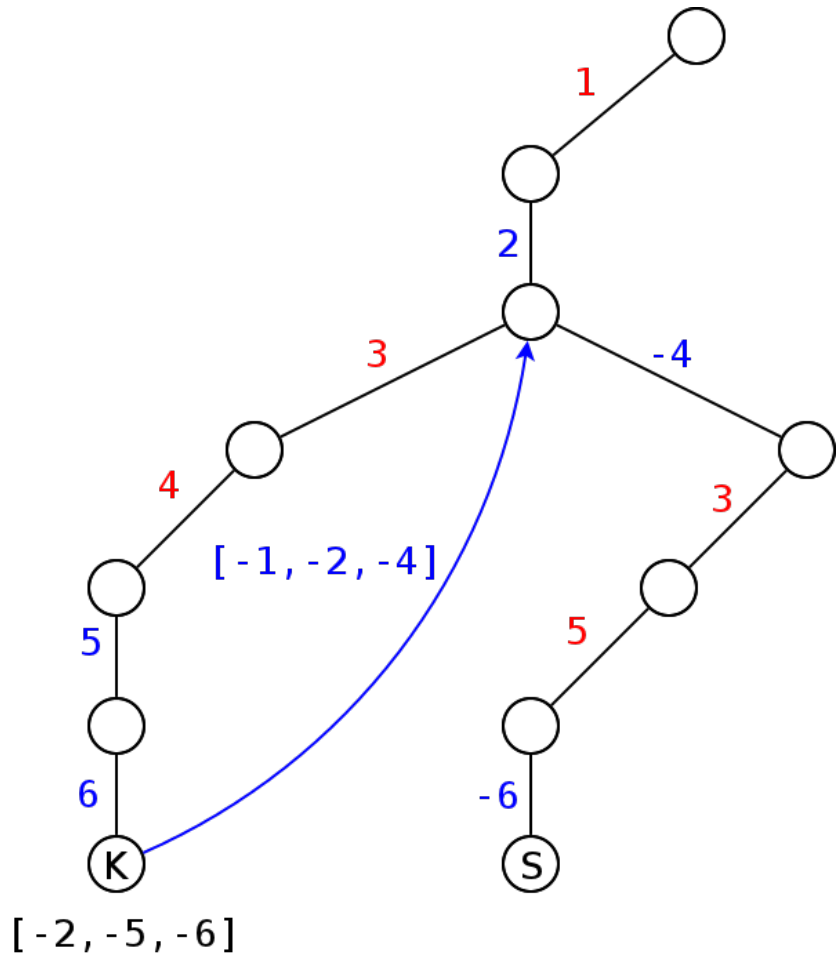
## Conflict Driven Clause Learning (CDCL)

- Konflikte werden analysiert
  - Anstatt Backtracking → Backjumping
  - Mehrere Entscheidungen können rückgängig gemacht werden
  - Aus Konflikt wird eine Klausel gelernt

# Lösen des SAT Problems

## Conflict Driven Clause Learning (CDCL)

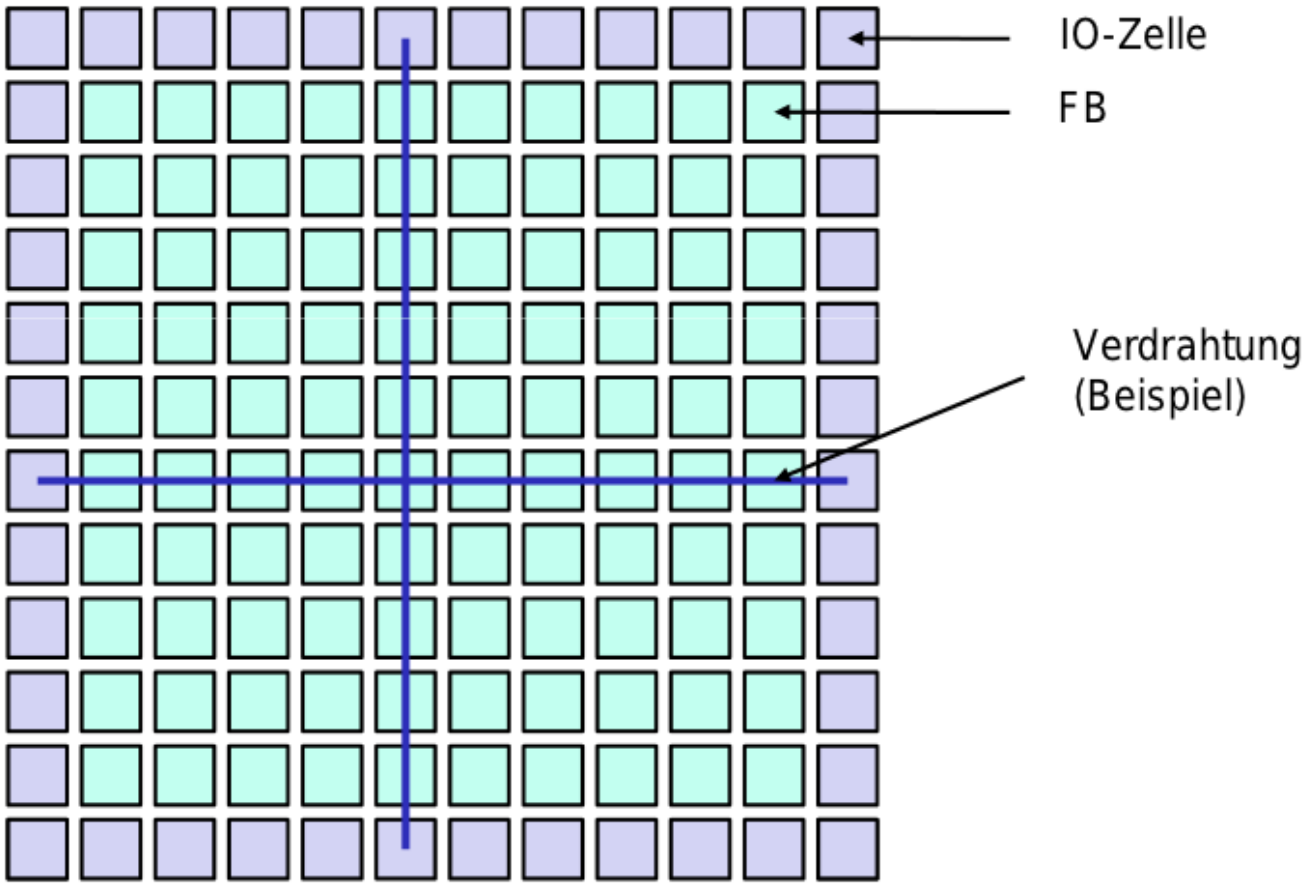
- F=⟨ [1, 3], [¬2, ¬5, ¬6], [¬1, ¬4, 6], [¬1,¬2,¬4, 5], [¬1, 2]⟩



# FPGA - Field Programable Gate Array

- Funktionsblöcke als Matrix angeordnet
  - Multiplexer basiert
  - Look up table basiert (LUT)
- I/O Zellen als spezielle Funktionsblöcke
- Programmierbare Verbindungen zwischen Funktionsblöcken
- Spezielle Hard Macros
  - Multiplizierer, Ethernet, Speicher Kontroller etc.
- zusätzlich BRAM auf FPGA integriert (Block-RAM)

# FPGA - Field Programable Gate Array



# Hardware SAT Solver

## Unterteilung nach Merkmalen

- Algorithmus
  - DPLL
  - CDCL
  - statische / dynamische Heuristiken

# Hardware SAT Solver

## Unterteilung nach Merkmalen

- Algorithmus
  - DPLL
  - CDCL
  - statische / dynamische Heuristiken
- Programmierkonzept
  - Instanz spezifisch
  - Applikations spezifisch

# Hardware SAT Solver

## Unterteilung nach Merkmalen

- Algorithmus
  - DPLL
  - CDCL
  - statische / dynamische Heuristiken
- Programmierkonzept
  - Instanz spezifisch
  - Applikations spezifisch
- Ausführungskonzept
  - nur Hardware
  - Hardware / Software

# Hardware SAT Solver

## Die ersten Hardware SAT Solver

- Viele Instanz spezifische SAT Solver im Zeitraum von 1996 bis 2000
  - Nur kleine Probleme lösbar (bis zu 100 Variablen)
  - Oft lange synthese Zeiten
  - Keine neueren Resultate im Vergleich zu moderne Software Solvern vorhanden

# Hardware SAT Solver

## Moderne Hardware SAT Solver

- Löser von Mona Safar et al. (2007)
  - nur Hardware
  - Applikation spezifisch
  - 100 Variablen und 200 Klauseln in 3 KNF

# Hardware SAT Solver

## Moderne Hardware SAT Solver

- Löser von John D. Davis et al. (2008)
  - Hardware / Software
    - Unit propagation wird in Hardware durchgeführt
    - Entscheidungen, Konfliktanalyse etc. in Software
  - Applikations spezifisch
  - 64k Variablen, 64k bis 176k Klauseln in 9 KNF
    - Werden in lokalem BRAM gespeichert
  - 7x to 30x speedup im Vergleich zu Zchaff

# Hardware SAT Solver

## Moderne Hardware SAT Solver

- Löser von Leopold Haller et al. (2010)
  - nur Hardware + DRAM Anbindung
    - Klauseln und Watcher Listen werden in DRAM gespeichert
    - Partielle Interpretation wird in lokalem BRAM gespeichert
  - Applikation spezifisch
  - 1M variablen und 70M Klauseln in 24 KNF
  - Keine experimentellen Resultate

# Hardware SAT Solver

## Vor und Nachteile einer FPGA Lösung

- Vorteile
  - Massiv parallele Ausführung
  - Keine Steuerbefehle nötig
- Nachteile
  - Niedriger Takt (typisch sind 20 - 500 Mhz)
  - Kommunikation zwischen FPGA und Host Rechner nötig

# Zu lösende Aufgaben

- Algorithmus: Welcher Algorithmus zum lösen von SAT eignet sich am besten ?
- Kommunikation: Wie schnell kann zwischen Host Rechner und FPGA kommuniziert werden ?
- Speicher: Welche SAT Probleme können gelöst werden ?
- Parallelität: Wie kann die massive Parallälität eines FPGAs ausgenutzt werden ?

Vielen dank für Ihre Aufmerksamkeit